Universidad San Carlos de Guatemala
Facultad de Ingeniería
Escuela de Ciencias y Sistemas
Introducción a la Programación y Computación 2

Inga. Claudia Liceth Rojas Morales
Ing. Marlon Antonio Pérez Türk
Ing. William Estuardo Escobar Argueta
Ing. José Manuel Ruiz Juárez
Ing. Edwin Estuardo Zapeta Gómez

Tutores de curso:
Oscar René Jordán Orellana, Juan Pablo Osuna de León
Astrid Edith Hernández González, Henry Adolfo Gálvez
César Dionicio Sazo Mayen, Oscar Alejando Rodríguez
Calderón Edwar Everaldo Zacarias, Javier Estuardo Lima
Mónica Raquel Calderón Muñoz, Marco Antonio López Grajeda

PROYECTO 3

OBJETIVO GENERAL

Desarrollar una solución integral que implemente un API que brinde servicios utilizando el Protocolo HTTP bajo el concepto de programación orientada a objetos (POO) y el uso de bases de datos.

OBJETIVOS ESPECÍFICOS

- Implementar un API a través de lenguaje Python que pueda ser consumida utilizando el protocolo HTTP.
- Utilizar el paradigma de programación orientada a objetos para construir software.
- Utilizar bases de datos para almacenar información de forma persistente.
- Utilizar archivos XML como insumos para la comunicación con el API desarrollado.
- Utilizar expresiones regulares para extraer contenido de texto.

ENUNCIADO

Una empresa de software le ha solicitado construir un software que pueda ser consumido desde Internet como un servicio. Este software recibirá un mensaje de la bitácora del software principal de la compañía y producirá una serie de información estadística relacionada. El mensaje que la bitácora enviará contendrá la siguiente información:

```
FECHA: dd/mm/yyyy
USUARIO: correo electrónico del usuario que genera el error
AFECTADO: lista de correos electrónicos separados por coma
ERROR: código numérico: descripción del error
```

El programa a desarrollar, luego de recibir el mensaje antes mencionado deberá almacenar la información necesaria en un archivo XML que permitirá mostrar la siguiente información:

```
FECHA: dd/mm/yyyy
 Cantidad total de mensajes recibidos en esta fecha
 Listado de usuarios distintos que reportaron mensajes
 Usuario1 cantidad mensajes generados
 Usuario2 cantidad mensajes generados
 …
 Listado de usuarios afectados
 UsuarioX1
 UsuarioX2
 …
 Listado de errores distintos reportados
 Código numérico del error: cantidad de mensajes recibidos
 Código numérico del error: cantidad de mensajes recibidos
 …
FECHA: dd/mm/yyyy
 …
```

OJO: *La sumatoria de los mensajes generados en el listado de usuarios que reportaron el mensaje y la sumatoria de los errores reportados deben cuadrar con la cantidad de mensajes recibidos en una fecha dada.*

ARCHIVOS DE ENTRADA Y SALIDA

Archivo de entrada

Si el archivo de entrada posee un error de sintaxis con respecto a la estructura XML entonces el evento con el error será ignorado y se continuará con el análisis de los eventos restantes. Dentro de la etiqueta <eventos> pueden venir 1 o más etiquetas <evento>.

```
<EVENTOS>
   <EVENTO>
      Guatemala, 15/01/2021
      Reportado por: <”Nombre Empleado 1” xx@ing.usac.edu.gt>
      Usuarios afectados: aa@ing.usac.edu.gt, <bb@ing.usac.edu.gt>
      Error: 20001 - Desbordamiento de búfer de memoria RAM
      en el servidor de correo electrónico.
   </EVENTO>
   …
</EVENTOS>
```

Archivo de salida

Nombre del archivo: estadistica.xml

```
<ESTADISTICAS>
   <ESTADISTICA>
      <FECHA> 15/01/2021 </FECHA>
      <CANTIDAD_MENSAJES> 3 </CANTIDAD_MENSAJES>
      <REPORTADO_POR>
         <USUARIO>
            <EMAIL> xx@ing.usac.edu.gt </EMAIL>
            <CANTIDAD_MENSAJES> 1 </CANTIDAD_MENSAJES>
         </USUARIO>
         <USUARIO>
            <EMAIL> yy@ing.usac.edu.gt </EMAIL>
            <CANTIDAD_MENSAJES> 2 </CANTIDAD_MENSAJES>
         </USUARIO>
      </REPORTADO_POR>
      <AFECTADOS>
         <AFECTADO> aa@ing.usac.edu.gt </AFECTADO>
         <AFECTADO> bb@ing.usac.edu.gt </AFECTADO>
         …
      </AFECTADOS>
      <ERRORES>
         <ERROR>
            <CODIGO> 20001 </CODIGO>
            <CANTIDAD_MENSAJES> 2 </CANTIDAD_MENSAJES>
         </ERROR>
         <ERROR>
            <CODIGO> 20002 </CODIGO>
            <CANTIDAD_MENSAJES> 1 </CANTIDAD_MENSAJES>
```

```
        </ERROR>
        ...
    </ERRORES>
</ESTADISTICA>
...
</ESTADISTICAS>
```

ARQUITECTURA

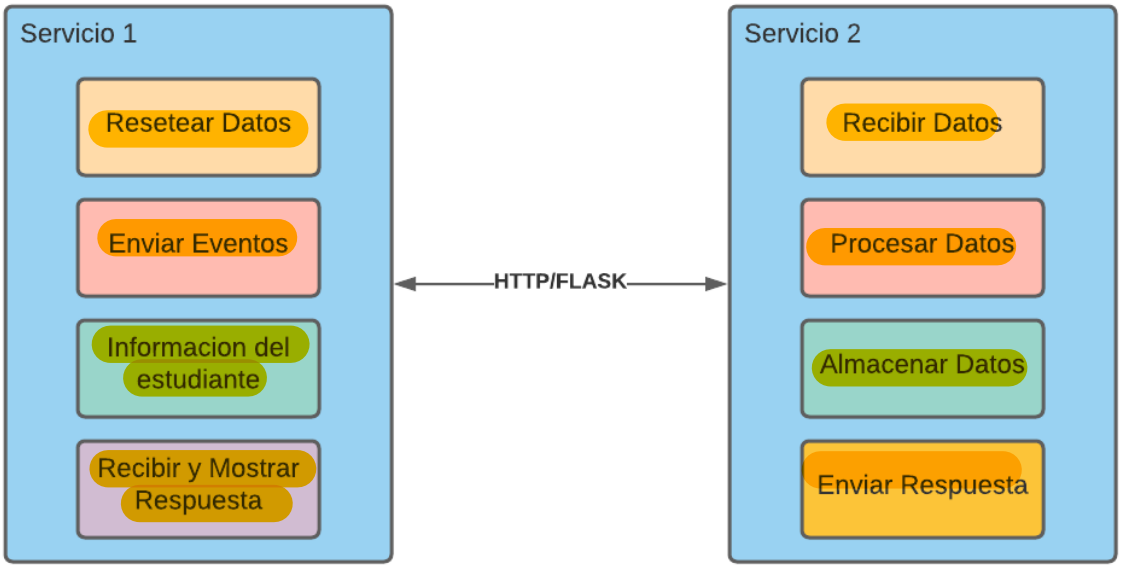

Figura 1: Arquitectura de la aplicación

Servicio 1 - Frontend

Este servicio consiste en una aplicación Web y consistirá en un simulador de la aplicación principal, contendrá únicamente las funcionalidades necesarias para testear el buen funcionamiento de la Api (Servicio 2), en esta aplicación se podrán mostrar los eventos que se procesarán y los datos estadísticos que fueron almacenados en la base de datos XML de salida).

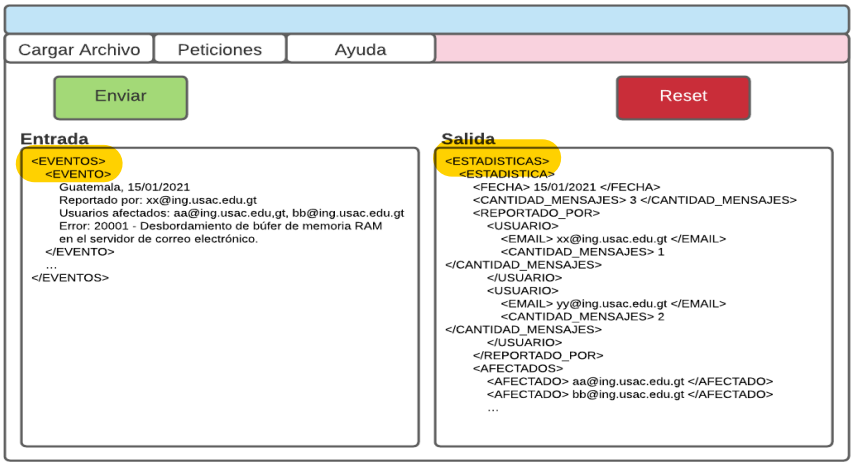

Figura 2: Prototipo de interfaz

Componentes:

- **Cargar Archivo:** Se desplegará una pantalla para gestionar la carga de los archivos de entrada con extensión .xml con uno o varios eventos. Se especifica en la sección de archivos de entrada y salida.
- **Peticiones:** En este apartado se debe de tener las siguientes opciones:

    - **Consultar Datos:** Al seleccionar esta opción se deben de consultar los datos almacenados en el archivo estadisticas.xml y se mostrarán los datos en el recuadro de texto de salida.
    - **Filtrar información por fecha y usuario que reporta:** Al seleccionar esta opción se podrá elegir la fecha por la cual se requiere filtrar y se debe de **mostrar gráficamente** los usuarios que reportaron errores en esa fecha.
    - **Filtrar por fecha y código de error:** Al seleccionar esta opción se podrá ingresar un código de error, se deberá **presentar gráficamente** el total de mensajes que contienen ese código por cada fecha en donde se hizo el reporte de dicho error.

- **Ayuda:** desplegará 2 opciones, una para visualizar información del estudiante y otra para visualizar la documentación del programa.
- **Botón Enviar:** Enviará los eventos del recuadro de texto a la Api para su posterior procesamiento.
- **Botón Reset:** Este botón mandará la instrucción a la Api para devolver al estado inicial la Api, es decir sin datos.

Servicio 2 - Backend

Este servicio consiste en una API que brindara servicios utilizando el protocolo HTTP, su funcionalidad principal es procesar los datos recibidos del servicio 1, luego de procesar los datos es necesario que estos sean almacenados en un archivo xml, este archivo está especificado en la sección de archivos de entrada y salida, este servicio también tiene la funcionalidad de devolver los datos que fueron almacenados para que sean mostrados como anteriormente se indica.

NOTA: Durante la calificación de este proyecto, el Servicio 2 podrá ser consumido desde otro cliente, por ejemplo, postman.

CONSIDERACIONES

En los eventos del archivo de entrada debe descartarse cualquier información que no sea necesaria para la elaboración de las estadísticas, por ejemplo, si los correos electrónicos vienen entre símbolos < y >, o traen nombres entre comillas, solamente debe presentar los correos electrónicos. Si las líneas de Reportado por, afectados, error o fecha, traen otra información, no se considerará un error, el objetivo es descartar cualquier información no relevante y obtener los datos necesarios para la estadística.

Debe utilizarse versionamiento para el desarrollo del proyecto. Se utilizará la plataforma Github en la cual se debe crear un repositorio en el que se gestionará el proyecto. Se deben realizar 4 releases o versiones del proyecto (se recomienda realizar una por

semana del tiempo disponible). Se deberá agregar a su respectivo auxiliar como colaborador del repositorio. El último release será el release final y se deberá de realizar antes de entregar el proyecto en la fecha estipulada.

DOCUMENTACIÓN

Para que el proyecto sea calificado, el estudiante deberá entregar la documentación utilizando el formato de ensayo definido para el curso. En el caso del proyecto, el ensayo debe tener entre 4 y 7 páginas de contenido, este máximo no incluye los apéndices o anexos donde se pueden mostrar modelos y diseños utilizados para construir la solución.

RESTRICCIONES

- Solo se permitirá la utilización de los IDEs discutidos en el laboratorio.
- Uso obligatorio de programación orientada a objetos (POO).
- El nombre del repositorio debe de ser **IPC2_Proyecto3_#Carnet.**
- El estudiante debe entregar la documentación solicitada para poder optar a la calificación.
- Se calificará de los cambios realizados en el cuarto release. Los cambios realizados después de ese release no se tomarán en cuenta.
- Para dudas concernientes al proyecto se utilizarán los foros en UEDI de manera que todos los estudiantes puedan ver las preguntas y las posteriores respuestas.
- **NO HABRÁ PRÓRROGA.**

ENTREGA

- La entrega será el **7 de mayo** a las 23:59 como máximo.
- La entrega será por medio de la UEDI.
- La documentación debe estar subida en el repositorio en una carpeta separada.
- Para entregar el proyecto en UEDI se deberá subir un archivo de texto con el link del repositorio.